6.1 INTRODUCTION TO FUNCTIONAL DEPENDENCIES

(MDU, May 2012, Dec. 2009)

Let R be a relation and A & B be the sub-sets of attribute of R, then B is said to be functionally dependent on A if, given A we can precisely determine B.

If B is functionally dependent on A then it is denoted as $A \rightarrow B$.

Now consider the following relation

Emp_Project
(EId, EName, Proj_No, Proj_Name, Proj_LOC, Hours)

The following FD's exists in above relation

- EId → EName

Given Id of an employee we can find out the name of the employee corresponding to that Id.

- Proj_No → Proj_Name, Proj_LOC

Given project number, we can find out the name of the project corresponding to that project number. Similarly we can find project LOC from Proj_No.

- {EId, Proj_No} → Hours

Given Employee Id and project number on which an employee is working, we can find out number of hours worked by each employee on a particular project.

- EName $\rightarrow\!\!\!\!\!\times$ EId

EName does not functionally determine EId because there may be two employee's with the same name.

Fully Functional Dependency

Let R be a relation, then an attribute B is said to be fully functionally dependent

on attribute A if it is functionally dependent on A and not functionally dependent on any proper sub-set of A.

The following FD

$$\{EId, Proj\_No, Proj\_LOC\} \rightarrow Hours$$

is not fully functionally dependent on the set on left hand side because number of hours worked by an employee on a project can also be determined by a sub-set of set on left side of FD.

$$\{EId, Proj\_No\} \rightarrow Hours$$

Hour is fully functionally dependent on EId and Proj_No because neither EId nor Proj_No alone can determine number of hours worked by an employee on a particular project.

Trivial Functional Dependency

(GGSIPU, 2011)

An FD is trivial if and only if the right hand side attribute(s) is a sub-set of left hand side attribute(s), i.e.

$A \rightarrow B$, where $B \subseteq A$

***Example*:**

Customer_name → Customer_name
{Customer_name, loan_number} → Customer_name

Non-Trivial Functional Dependency

(GGSIPU, 2008, 2011)

An FD in which at least one attribute on right side of FD not contained in left side attributes of FD i.e. $A \rightarrow B$, $B \not\subseteq A$ (B is not a sub-set of A)

Example: {EId, Project_No} → Hours

Here Hours $\not\subseteq$ {EId, Project_No}.

6.2 CLOSURE OF A SET OF FUNCTIONAL DEPENDENCY

(GGSIPU, 2010)

The set of all FDs that are logically implicit by a given set s of FDs is called closure of S and is denoted by S^+.

Example: If $A \rightarrow B$ and $B \rightarrow C$, then $A \rightarrow C$

S^+ can be found out by the following Armstrong inference rules.

(MDU, May 2011; UPTU 2011-12)

1. **Reflexivity Rule:** If A is a set of attributes and $B \subseteq A$, then $A \rightarrow B$ holds
2. **Augmentation Rule:** If $A \rightarrow B$, C is a set of attributes then $AC \rightarrow BC$.
3. **Transitivity Rule:** If $A \rightarrow B$ and $B \rightarrow C$, then $A \rightarrow C$.
4. **Union Rule:** If $A \rightarrow B$ and $A \rightarrow C$, then $A \rightarrow BC$.
5. **Self-determination Rule:** $A \rightarrow A$
6. **Decomposition Rule:** If $A \rightarrow BC$, then $A \rightarrow B$ and $A \rightarrow C$.
7. **Pseudotransitivity Rule:** If $A \rightarrow B$ and $CB \rightarrow D$, then $AC \rightarrow D$

Example: Suppose we are given a relation R with attribute A, B, C, D, E, F and FDs **(GGSIPU 2013)**

$A \rightarrow BC$
$B \rightarrow E$
$CD \rightarrow EF$

Prove $AD \rightarrow F$

Solution:

$A \rightarrow BC$ (Given)
$A \rightarrow C$ (Decomposition)
$AD \rightarrow CD$ (Augmentation)
$CD \rightarrow EF$ (Given)
$AD \rightarrow EF$ (Transitivity)
$AD \rightarrow F$ (Decomposition)
Hence proved.

Example: Suppose we are given a relation R with attribute A, B, C, D, E, F, G and FDs

$A \rightarrow B$
$ABCD \rightarrow E$
$EF \rightarrow G$

Prove $ACDF \rightarrow G$

Solution:

$A \rightarrow B$
$ABCD \rightarrow E$ (Given)
$ACD \rightarrow E$ (Pseudotransitivity)
$ACDF \rightarrow EF$ (Augmentation)
$ACDF \rightarrow G$ ($EF \rightarrow G$ (Transitivity))
Hence proved.

6.3 CLOSURE OF A SET OF ATTRIBUTES

Given a relation R, a set of Z of attributes of R and a set S of FDs that hold for R,

e define closure of Z denoted by Z^+ as the set of attribute that are functionally etermined by Z under S.

Algorithm

closure = Z;
while (closure do not change)
do
for each FD $X \rightarrow Y$ in S
do
if $X \subseteq$ closure, then
closure = closure $\cup$ Y;

Example: Suppose a relation R with attribute A, B, C, D, E, F and FDs

$A \rightarrow BC$
$E \rightarrow CF$
$B \rightarrow E$
$CD \rightarrow EF$

Compute the closure of set of attribute {A, B}

(i) Initialise closure $\{A, B\}^+$ to {A, B}. ... (1)
(ii) For each FD check if left side attribute is a sub-set of $\{A, B\}^+$, if yes then add right side attribute which ever not yet included in $\{A, B\}^+$
$A \rightarrow BC$, Add C (B already present)
$AE \rightarrow CF$, Add nothing
$B \rightarrow E$, Add E
$CD \rightarrow EF$, Add nothing ... (2)

$\{A, B\}^+ = \{A, B, C, E\}$.

(iii) Repeat step 2 until result keep on changing ... (3)
$\{A, B\}^+ = \{A, B, C, E\}$ ... (4)
$\{A, B\}^+ = \{A, B, C, E, F\}$ ... (5)
$\{A, B\}^+ = \{A, B, C, E, F\}$

Since (4) and (5) are same so, it is the required closure.

Example: Given a relation R with attribute A, B, C, D, E, F, G and FDs.

$A \rightarrow B$
$BC \rightarrow DE$
$AEF \rightarrow G$

Compute $\{A, C\}^+$.

Solution: $\{A, C\}^+ = \{A, C\}$
$= \{A, B, C\}$

$= \{A, B, C, D, E\}$
$= \{A, B, C, D, E\}$

6.4 EQUIVALENCE OF SETS OF FUNCTIONAL DEPENDENCIES

Let $S1^+$ and $S2^+$ be two sets of FDs. If S1+ is a sub-set of S2+ i.e. every FD implied by S1 is implied by S2 then S2 is said to be the cover for S1.

Now, if S2 is a cover for S1 and S1 is a cover for S2 i.e. $S1^+ = S2^+$, then S1 and S2 are equivalent.

6.5 CANONICAL COVER/MINIMAL COVER/IRREDUCIBLE SET OF FUNCTIONAL DEPENDENCIES

(GGSIPU 2013, 2012, 2007)

For every set of functional dependencies, if a minimal simplified set that is equivalent to a given set of functional dependencies can be generated, then the effort spent in checking for violation can be reduced on each update. A minimal simplified set has the same closure as that of the given set and database that satisfies the simplified set of functional dependencies will also satisfy the original given set F of functional dependencies. For example consider the functional dependencies $AB \rightarrow CD$ and $A \rightarrow D$. If we remove attribute D from right side of $AB \rightarrow CD$, then it does not affect closure F because $AB \rightarrow C$ and $A \rightarrow D$ implies $AB \rightarrow CD$ by union rule. Therefore we can replace functional dependency $AB \rightarrow CD$ and $A \rightarrow D$. Verifying this new set require less effort then original set.

The attribute C is extraneous in functional dependency $AB \rightarrow CD$ because closure of new set will be same as old set. An extraneous attribute is formally defined as follows:

Consider functional dependency $X \rightarrow A$ in given set F of functional dependencies.

- Attribute Y is extraneous in X if $Y \in X$ and F logically implies $(F - \{ X \rightarrow A \}) \cup \{ (X - Y) \rightarrow A \}$

Now we give an algorithm to find a canonical or minimal cover of a set F of FDs

Input: A set F of FDs

Output: A canonical or minimal cover M of set F

1. Replace each FD: $X \rightarrow A_1 A_2 \ldots\ldots\ldots A_k$ in F by k FDs, with only single attribute on right side using decomposition rule. $X \rightarrow A_1$, $X \rightarrow A_2$, ..., $X \rightarrow A_k$.
2. Remove individual FDs that are inessential in F. An FD is inessential if its removal from F has no effect on F^+.

```
for (all FD X → A in F)
{
        J= F – { X → A} ;     // remove a FD from F
```

```
        Find X+ under J;          // find closure of X without FD X→A
        if (A ∈ X+)               // X→A is implied by J
        F = F – { X → A};         // FD X→A can be removed from F
}
```

3. Successively remove extraneous attribute from left hand side, as long as the result does not change F^+.

```
H = F                             // save original F
for (all FD X → A in F with more than one attribute on left side)
    {
    for (all Y ∈ X )
      {
      Z = X – {Y}                 // remove one attribute from left side
      J = ( F – { X→A}) ∪ (Z → A);
      Find Z+ under F and Z+ under J;
      if (Z+ under F == Z+ under J)
        {
        Replace X → A by Z → A;  //remove extraneous attribute Y from X
        }
      }
    }
  if(F != H)
  Repeat step 2 and then go to step 4;
```

4. From remaining set of FDs with equal left side, use union rule to create an equivalent set of FDs M where all left side are unique.

Example: Find out the canonical cover for set F of FD, $A \rightarrow C$, $AC \rightarrow D$, $E \rightarrow ADH$

Step 1: Apply decomposition rule to obtain equivalent set with single attribute on right side.

$F = \{ A \rightarrow C , AC \rightarrow D , E \rightarrow A , E \rightarrow D , E \rightarrow H \}$

Step 2: Remove any redundant FDs

- Consider $A \rightarrow C$, compute $\{A\}^+$ without FD $A \rightarrow C$
  $\{A\}^+ = \{A\}$
  Since C is not in A^+, FD $A \rightarrow C$ is essential.
  $F = \{A \rightarrow C, AC \rightarrow D, E \rightarrow A , E \rightarrow D , E \rightarrow H\}$
- Consider $AC \rightarrow D$, compute $\{AC\}^+$ without FD $AC \rightarrow D$
  $\{AC\}^+ = \{AC\}$
  Since D is not in $\{AC\}^+$, FD $AC \rightarrow D$ is essential.
  $F = \{A \rightarrow C, AC \rightarrow D, E \rightarrow A, E \rightarrow D, E \rightarrow H\}$
- Consider $E \rightarrow A$, compute $\{E\}^+$ without $E \rightarrow A$
  $\{E\}^+ = \{EDH\}$
  Since A is not in $\{E\}^+$, FD $E \rightarrow A$ is essential
  $F = \{A \rightarrow C, AC \rightarrow D, E \rightarrow A, E \rightarrow D, E \rightarrow H\}$
- Consider $E \rightarrow D$, compute $\{E\}^+$ without $E \rightarrow D$

$\{E\}^+ = \{EAHCD\}$

Since D is in $\{E\}^+$, FD $E \rightarrow D$ is not essential, so remove $E \rightarrow D$

$F = \{A \rightarrow C, AC \rightarrow D, E \rightarrow A, E \rightarrow H\}$

- Consider $E \rightarrow H$, compute $\{E\}^+$ without $E \rightarrow H$

$\{E\}^+ = \{EACD\}$

Since H is not in $\{E\}^+$, FD $E \rightarrow H$ is essential.

$F = \{A \rightarrow C, AC \rightarrow D, E \rightarrow A, E \rightarrow H\}$

Step 3: Remove extraneous attribute if any for FD they have more than one attribute on left side

$F = \{A \rightarrow C, AC \rightarrow D, E \rightarrow A, E \rightarrow H\}$

- Consider $AC \rightarrow D$

For A compute $\{C\}^+$ under set $J = \{A \rightarrow C, C \rightarrow D, E \rightarrow A, E \rightarrow H\}$

$\{C\}^+ = \{CD\}$

For A compute $\{C\}^+$ under set $F = \{A \rightarrow C, AC \rightarrow D, E \rightarrow A, E \rightarrow H\}$

$\{C\}^+ = \{C\}$

$\{C\}^+$ under J is not equal to $\{C\}+$ under F therefore so A is not an extraneous attribute.

For C compute $\{A\}^+$ under set $J = \{A \rightarrow C, A \rightarrow D, E \rightarrow A, E \rightarrow H\}$

$\{A\}^+ = \{ACD\}$

For A compute $\{A\}^+$ under set $F = \{A \rightarrow C, AC \rightarrow D, E \rightarrow A, E \rightarrow H\}$

$\{A\}^+ = \{ACD\}$

$\{A\}^+$ under J is equal to $\{A\}+$ under F, therefore C is an extraneous atribute.

Replace $AC \rightarrow D$ with $A \rightarrow D$

$F = \{A \rightarrow C, A \rightarrow D, E \rightarrow A, E \rightarrow H\}$

Step 4: Using union rule combine FDs having same left side.

$M = \{A \rightarrow CD, E \rightarrow AH\}$

6.6 DECOMPOSITION

(GGSIPU, 2011)

Decomposition refers to breaking down of one relation into multiple relations.

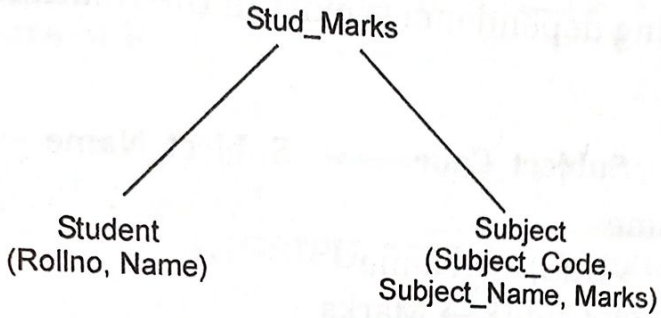
Consider the following relation:

Stud_Marks

Rollno	Name	Subject_Code	Subject_Name	Marks
25	Ankit	CS-21	DBMS	92
25	Ankit	CS-20	OOPS	85
25	Ankit	CS-22	Data Structure	76
26	Saurabh	CS-21	DBMS	86
26	Saurabh	CS-20	OOPS	65

Fig. 6.1

The relation above contains redundant information. The rollno. and name is epeated for each subject. So, to remove the redundancy we divide the relation nto two new relations, one contains the student information and other contains ubject and marks information.



Now after this decomposition, student Rollno. and Name will not be repeated. But we have lost some information that which student has obtained how much marks and in which subject, so it is a bad decomposition.

Loosy Decomposition

Suppose a relation R is decomposed into two new relation R1 and R2. If on recomposition of R1 and R2, the original relation R cannot be obtained due to loss of information, than the decomposition is called loosy decomposition. In the previous example, relations student and Marks cannot be recomposed to original relation Stud_Marks because on decomposition we have lost the information that which student has got how many marks and in which subject.

Lossless Join Decomposition

(GGSIPU, 2013, 2005; MDU, May 2011; UPTU 2006-07)

When on decomposition of a relation, recomposition can be done without loss of information then the decomposition is called lossless decomposition. A decomposition of R to R_1, R_2, R_3, ... R_n is said to have lossless join property with respect to the set of dependencies S on R if, for every relation state r of R that satisfies S, following condition hold.

$$\pi_{R_1}(r) \bowtie \pi_{R_2}(r) \ldots \bowtie r_{R_n}(r) = r$$

Here π is a projection operation and $\bowtie$ is a natural join operation of all decomposed relations.

A simple way to test whether the decomposition is lossless is given below:

Let R be a relation and F be a set of FDs that hold over R. The decomposition of R into relation with attribute sets R_1 and R_2 is lossless if and only if F^+ contains either the FD.

$R_1 \cap R_2 \rightarrow R_1$ or
$R_1 \cap R_2 \rightarrow R_2$

i.e. attribute common to R_1 and R_2 must contain key for either R_1 or R_2.

Dependency Preserving Decomposition

This property states that each FD in set S of FDs appears directly in one of the decomposed relations or could be inferred from any FD that appears in any decomposed relations.

Example: Following dependencies exist in the relation Stud_Marks

Stud_Marks

Rollno	Name	Subject_Code	Subject_Name	Marks

Roll No → Name
Subject_code → Subject_Name
{Rollno, Subject_Code} → Marks

FDs left after decomposing the relation into following relations are

Student

Rollno	Name

Marks

Marks	Subject_Code	Subject_Name

Rollno → Name
Subject_Code → Subject_Name

We see that one FD is lost, so the decomposition is loosy.

If the following decomposition is done

Student

Rollno	Name

Subject

Subject_Code	Subject_Name

Marks

Rollno	Subject_Code	Marks

Then the FDs are
Rollno → Name
Subject_Code → Subject_Name
Rollno, Subject_Code → Marks

We see that no FDs are lost after decomposing so the decomposition is lossless decomposition.

In other words the decomposition of a relation schema R into $R_1, R_2 \ldots R_m$ with respect to set of functional dependencies F is dependency preserving if

$$\pi_{R_1}(F) \cup \pi_{R_2} \cup \ldots \left(\pi_{R_m}(F)\right) = F^+$$

where $\pi_{R_i}(F)$ is the projection of F on R_i.

A decomposition of relation scheme R with FDs F into R_1 and R_2 with attribute set X and Y respectivelyh is dependency preserving. if

$$(F_X \cup F_Y)^+ = F^+$$

F_X = Projection of X.
F_Y = Projection of Y.

that is, if closure of union of dependencies in F_X and F_Y is done, we get back all dependencies in closure of F.

6.7 NORMALISATION

(GGSIPU, 2006, 2008, 2009, 2011; KU Dec 2005; UPTU 2011-12; PTU-2011, 2012)

Normalisation is a process of decomposing a relation into various, smaller relations that contains minimum or no redundancy.

Normalisation Avoids

(MDU, May 2011)

1. **Duplication of Data**
2. **Insert Anomaly:** A record about an entity cannot be inserted into the table without first inserting information about another entity.

Consider the following relation

Student

ID	Name	Dept. No.	Dept. Name
1	Gaurav	1	CSE
2	Shivangi	1	CSE
3	Hitesh	2	IT
4	Mohit	3	ECE
5	Deepak	2	IT

Fig. 6.1: Student Table

We cannot add a department ME unless there is a student in this department.

3. **Delete Anomaly:** A record cannot be deleted without deleting a record about a related entity.
   Example: We cannot delete department CSE until information about all students that belong to CSE is deleted.
4. **Update Anomaly:** We cannot update information without changing information in many other places. If this is not done then data would become inconsistent. **(GGSIPU 2013)**
   Example: If we want to change department number of IT from 2 to 4 then, it should be updated for all students that belong to IT branch.

6.7.1 First Normal Form

A relation is said to be in first normal form if the domain of an attribute have only atomic values. This means that the relation does not contain any repeating groups of columns i.e. only one value is associated with each attribute and the value is not a set of value.

Example: Consider the following relation

Student

RollNo	Name	Course_Code	Duration	Subject_Code	Subject_Name	Teacher
1	Anuj	A-26	2	CS-51 CS-52	DBMS Compiler	Shalini Radhika
2	Maya	A-49	3	CS-51 CS-55	DBMS OOPS	Shalini Mukesh
3	Pankaj	A-52	4	CS-59 CS-55 CS-54	Java OOPS OS	Komal Mukesh Kapil

Fig. 6.2: Unnormalized Student Relation

In the above relation we see that for each value of RollNo the attribute Subject_Code, Subject_Name and Teacher have multiple value. So, Subject_Code, Subject_Name and Teacher attributes are not atomic and hence not in INF.

To convert this relation into INF we remove the attributes that violates INF and place them in a separate relation along with the primary key of the original relation.

After decomposition the two new relation formed are:

- Student (RollNo, Name, Course Code, Duration)
- Subject (RollNo, Subject_Code, Subject_Name, Teacher)

The primary key for the new relation Subject will be a composite key {RollNo, Subject_Code} because each combination of attributes RollNo and Subject_Code uniquely identifies each subject name and teacher.

Student 1

RollNo	Name	Course_Code	Duration
1	Anuj	A 26	2
2	Maya	A 49	3
3	Pankaj	A 52	4

RollNo	Subject_Code	Subject_Name	Teacher
1	CS-51	DBMS	Shalini
1	CS-52	Compiler	Radhika
2	CS-51	DBMS	Shalini
2	CS-55	OOPS	Mukesh
3	CS-59	Java	Komal
3	CS-55	OOPS	Mukesh
3	CS-54	OS	Kapil

Fig. 6.3: Student1 and Subject Relation in 1 NF

We see that in above relation there are no repeating groups so both the relation are in INF.

6.7.2 Second Normal Form (PTU 2010)

A relation is said to be in 2 NF if it is in INF and every non-prime attribute of R is fully functionally dependent on primary key of R.

Partial Dependency - Let primary key X of a relation is a composite key and a FD $X \rightarrow Y$ exist. Now, if some attribute $A \in X$ can be removed from X and dependency still holds i.e. $(X - \{A\}) \rightarrow Y$ then a partial dependency exist.

In 2 NF we remove all partial dependencies from a relation if they exist. If primary key is a single attribute, then no partial dependencies need to be checked.

Now, the relations student and subject are in INF so first condition for 2 NF is satisfied.

Relation Student1 has a single primary key so no partial dependencies exist.

Relation Subject has a composite primary key so, we need to check for partial dependencies in this relation.

Subject

RollNo	Subject_Code	Subject_Name	Teacher

Following FDs exist in above relation:

FD1 RollNo, Subject_Code → Subject_Name, Teacher
FD2 Subject_Code → Subject_Name
FD3 Subject_Code → Teacher

Therefore, we see a partial dependency exist in following FDs

Subject_Code → Subject_Name
Subject_Code → Teacher

To remove these partial dependency we decompose the relation Subject into two new relations.

Student_Subject (*RollNo, Subject_Code*)

Subject_teacher (Subject_Code, Subject_Name, Teacher).

Now partial dependencies exist. After 2NF following three relations exist.

Student1 (RollNo, Name, Course_Code, Duration)
Student_Subject (RollNo, Subject_Code)
Subject_teacher (Subject_Code, Subject_Name, Teacher)

6.7.3 Third Normal Form

A relation R is said to be in 3NF if it is 2NF and no non-prime attribute of R is transitivity dependent on primary key of R.

Transitive dependency: Suppose in a relation R two FDs X → Y and Y → Z exists. This implies that Z is transitivity dependent on X (i.e. X → Z)

In 3NF we remove all the transitive dependencies from a relation if they exist

In Student1 relation the FDs are

RollNo → Name, Course_Code, Duration
Course_Code → Duration
The dependency RollNo → Duration is transitive through Course_Code.

- RollNo → Course_Code → Duration

To remove this dependency we decompose the Student relation to following relation.

- Student2 (RollNo, Name, Course_Code)
- Course (Course_Code, Duration)

There are no transitive dependencies in Student_Subject relation as there are no non-prime attributes. Also, there are no transitive dependencies in Subject_teacher relation because primary key Subject_Code determines Subject_Name and Teacher.

So, following relation exist after 3NF

Student2 (RollNo, Name, Course Code)
Course (Course_Code, Duration)
Student_Subject (RollNo, Subject_Code)
Subject_teacher (Subject_Code, Subject_Name, Teacher)

6.7.4 BCNF (Boyce-Codd Normal Form)

A relation schema R is said to be in BCNF if whenever a non-trivial FD X → A holds in R, then X is a super key of R. This means that for every FD that exist in a relation the only determinants (attribute on left side) are candidate keys.

Consider the following relation

Teach (RollNo, Course, Teacher)

If we assume that a student can have more than one subject, a teacher can teach only one subject and a subject can be taught by more than one teacher then following FD exists in above relation.

(i) {RollNo, Course} → Teacher
(i) Teacher → Course

In FD (i) the determinants {RollNo, Course) can be a candidate key for the relation.

In FD (ii) Teacher cannot be a candidate key for the relation. So, the relation is not in BCNF.

Therefore, to convert the relation into BCNF we decompose the relation teaches into following relations.

Teacher_Course (Teacher, Course)
Student_Teacher (RollNo, Teacher)

BCNF was proposed as a simpler form of 3 NF, but was found to be stricter than 3NF because every relation in BCNF is also in 3NF, but a relation in 3 NF is not necessary in BCNF. The difference, between 3NF and BCNF is that for a functional dependency A→B, 3NF allows this dependency in a relation if B is a primary key attribute and A is not a candidate key where as BCNF insist that for this dependency to remain in a relation, A must be a candidate key. **(GGSIPU 2007)**

Violation of BCNF may occur when:

- The relatin contains two or more composite candidate keys.
- The candidate key overlap, that is have at least one attribute in common.

(MDU, May 2009)

6.7.5 Fourth Normal Form

A relation is said to be in 4NF with respect to a set of dependencies S if for every non-trivial multi-valued dependency $X \rightarrow\rightarrow Y$ in S^+, X is a super key for R.

Multi-Valued Dependency: If in a relation R, having attributes X and Y and if for a given X, there can be multiple values of Y, then a multi-valued dependency exists between X and Y.

Consider the relation

Employee

EName	Proj_Id	Language
Rajeev	21	English
Rajeev	35	Hindi
Rajeev	21	Hindi
Rajeev	35	English
Amit	42	English

Fig. 6.4: Employee Relation

The relation employee is not in 4NF because non-trivial multi-valued dependencies EName →→ Proj_Id, EName →→ Language exists and EName is not a super key of employee relation

An employee may work on many project and may know many languages. Project taken and languages known are independent of each other. So, to capture this fact and remove multi-valued dependencies we decompose the relation into following two relations:

Emp_Proj

EName	Proj_ID
Rajeev	21
Rajeev	35
Amit	42

Emp_Language

EName	Language
Rajeev	English
Rajeev	Hindi
Amit	English

Fig. 6.5: Emp_Proj and Emp_Language Relation in 4NF

6.7.6 Fifth Normal Form

A relation is said to be in 5NF also called projection join normal form, if it is in 4NF and every non-trivial join dependency in the relation is implied by the candidate key of R.

Join Dependency: Let R be a relation and let A, B, ...Z be sub-sets of attributes of R, then we say that R satisfies the join dependency. *{A, B, ...Z} if and only if every legal value of R is equal to the join of its projection on A, B, ...Z.

5NF is concerned with those relations which can be divided into sub-relations but cannot be reconstructed. There are some relations that cannot be non-loss decomposed into two relations but can be non-loss decomposed into three relations.

Consider the following relation:

E-S-P

EName	Skill	Project_Id
Imran	Programmer	P2
Imran	Testing	P1
Abhishek	Programmer	P1
Imran	Programmer	P1

Fig. 6.6: E-S-P Relation

The above relation contains no non-trivial FDs or multi-valued dependencies so the relation is in 4NF. The relation E-S-P can be decomposed into three relations as shown below.

E-S

EName	Skill
Imran	Programmer
Imran	Testing
Abhishek	Programmer

S-P

Skill	Project_Id
Programmer	P2
Testing	P1
Programmer	P1

P-E

Project_Id	EName
P2	Imran
P1	Imran
P1	Abhishek

Fig. 6.7: E-S, S-P, P-E Relation in 5NF

Now, if we join two relations E-S and S-P, then it produce a copy of original E-S-P relation with one superiors tuple, which is shown below.

E-S-P

EName	Skill	Project_Id
Imran	Programmer	P2
Imran	Programmer	P1
Imran	Testing	P1
Abhishek	Programmer	P2 ⟶ Extra tuple
Abhishek	Programmer	P1

Fig. 6.8: E-S-P Relation with Extra Tuple

This shows that decomposition is loosy. But when all the three relations E_S, S_P and P_E are joined then we obtain the original relation E_S_P. Therefore, the new three relation obtained are in 5NF.

6.7.7 Domain Key Normal Form

A relation is in domain key normal form if and only if every constraint on the relation is a logical consequence of key constraints and domain constraints.

SOLVED EXAMPLES

Example 1: Prove the following **(GGSIPU 2010, 12)**

(i) If a relation is in BCNF then it is necessarily in 3NF.

(ii) There exist relations which are in 3NF but not in BCNF.

To prove (i) suppose that a relation $R(A_1, A_2 \ldots\ldots A_n)$ is in BCNF but it still has a transitive dependency $X \to Y$, $Y \not\to X$, $Y \to A_i$, for some i (i =1, 2....n), where X is a key and A_i is not in X since A_i is not a key attribute. A_i is not in Y since otherwise the dependency $Y \to A_i$ would have been trivial, contrary to the definition of transitive dependency the existence of a non-trivial functional dependency $Y \to A_i$ and the assumption that R is in BCNF imply that $Y \to A_j$ for all j-1,2...n, but then $Y \to X$ contrary to the assumption about the existence of the transitive dependency $X \to Y$, $Y \to X$, and $Y \to A_i$. This proves that a transitive

dependency cannot exit in a relation which is in BCNF and thus such a relation is also in 3NF.

To prove (ii) we simply take as an example the relation R(ABC) for which the functional dependencies AB→C and C→B hold. R is in 3NF but not in BCNF.

Example 2: Given a relation R {A, B, C, D, E, F, G} and set F of FDs

$$\begin{aligned} A &\rightarrow B \\ BC &\rightarrow D \\ BC &\rightarrow E \\ AEF &\rightarrow G \\ B &\rightarrow G \end{aligned}$$

Find super key and candidate key of the relation.

Solution. (i) To find super key from given FD combine all attribute present on left side of FD without repeating any attribute.

$\Rightarrow$ {A, B, C, E, F}

(ii) To find the candidate key from super key reduce the set of elements.

{A, B, C, E, F}
= {A, B, C, F} ($\because BC \rightarrow E$)
= {A, C, F} ($\because A \rightarrow B$)

hence, {A, C, F} is a candidate key of relation.

Example 3: Given a relation R (A, B, C, D, E) with set F of FDs

$$\begin{aligned} AB &\rightarrow C \\ CD &\rightarrow E \\ DE &\rightarrow B \end{aligned}$$

Determine whether AB and ABD is candidate key.

If closure of AB and ABD includes all attributes of R then, they are key otherwise not.

Solution. (i) Finding closure of AB

$$\begin{aligned} \{A, B\}^+ &= \{A, B\} \\ &= \{A, B, C\} \\ &= \{A, B, C\} \end{aligned}$$

Since $\{A, B\}^+$ does not contain D and E, it is not a key of R.

(ii) Finding closure of ABD

$$\begin{aligned} \{A, B, D\}^+ &= \{A, B, D\} \\ &= \{A, B, C, D\} && (\because AB \rightarrow C) \\ &= \{A, B, C, D, E\} && (\because CD \rightarrow E) \\ &= \{A, B, C, D, E\} \end{aligned}$$

So, A, B, D is a key for R since, $\{A, B, D\}^+$ contain all attributes of R.

Example 4: Suppose that we decompose schema **(GGSIPU, 2007, 2009)**

$$R = \{A, B, C, D, E\} \text{ into } \{A, B, C\} \text{ and } \{A, D, E\}$$

Show that this decomposition is a lossless decomposition, if following FD holds

$$A \rightarrow BC$$
$$CD \rightarrow E$$
$$B \rightarrow D$$
$$E \rightarrow A$$

Solution: A decomposition $\{R_1, R_2\}$ is lossless decomposition if

$$R_1 \cap R_2 \rightarrow R_1 \text{ or}$$
$$R_1 \cap R_2 \rightarrow R_2 \text{ or}$$
$$R1 \cap R_2 \rightarrow R_1 - R_2$$

$\Rightarrow R_1 \cap R_2$ forms a super key of either R_1 or R_2

If $R_1 = \{A, B, C\}$
$R_2 = \{A, D, E\}$
Then $R_1 \cap R_2 = A$
Now $A \rightarrow BC$ (given)
$\therefore$ A is a candidate key i.e. so, it is a lossless decomposition.

➲ Same question can also be solved by Matrix Method.

For this construct a matrix as shown below

	A	B	C	D	E
R_1					
R_2					

Put symbol 'a' in box for elements present in R1 and R2 respectively and 'bij' for those elements which are not present in R1 and R2 respectively.
R1 = {A, B, C}, R2 = {A, D, E}

	A	B	C	D	E
R_1	a	a	a	b_{14}	b_{15}
R_2	a	b_{22}	b_{23}	a	a

Now, apply the given FD and develop a new matrix after changing symbol b_{ij} to a_j if the element is functionally dependent.

Now, if any row contains all 'a' then it is a lossless decomposition otherwise lossy.

Apply $A \rightarrow BC$

	A	B	C	D	E
R_1	a	a	a	b	b
R_2	a	a_2	a_3	a	a

↓ ↓

Changed from b to a

Now, row R2 contains all a so, decomposition is lossless.

Example 5: Consider a Relation R = {A, B, C, D, E, F, G, H, I} and set F of FD **(GGSIPU, 2013, 2011, 2010, 2007)**

$$\{A, B\} \rightarrow C$$
$$A \rightarrow D, E$$
$$B \rightarrow F$$
$$F \rightarrow G, H$$
$$D \rightarrow I, J$$

Find the key for R and decompose R into 2NF and 3NF.

Solution. (i) Find key of R

$$\{AB^+\} = \{A, B, C, D, E, F, G, H, I, J\}$$
$$\{A^+\} = \{A, D, E, I, J\}$$
$$\{B^+\} = \{B, F, G, H\}$$
$$\{F^+\} = \{F, G, H\}$$
$$\{D^+\} = (D, I, J)$$

Hence AB is the key

(ii) Decomposing into 2NF

Determine attribute which are functional dependent on part of key i.e. A or B alone.

For this determine closure A^+ and B^+

$$A^+ = \{A\}$$
$$= \{A, D, E\}$$
$$= \{A, D, E, I, J\}$$
$$= \{A, D, E, I, J\}$$
$$B^+ = \{B\}$$
$$= \{B, F\}$$
$$= \{B, F, G, H\}$$
$$= \{B, F, G, H\}$$

Remove attribute that are functionally dependent on part of key and place them in separate relation.

$$R1 = \{\underline{A}, D, E, I, J\}$$
$$R2 = \{\underline{B}, F, G, H\}$$
$$R3 = \{\underline{A, B}, C\}$$

(iii) Decompose into 3NF. For this we need to find transitive dependencies

In R1

$A \rightarrow D$

$D \rightarrow I, J$ (given FD)

$\therefore A \rightarrow D \rightarrow I, J$

So, we remove transitive dependent attribute from R_1

$R4 = \{\underline{D}, I, J\}$

$R5 = \{\underline{A}, D, E\}$

In R2

$B \rightarrow F$

$F \rightarrow G, H$ (given FD)

$B \rightarrow F \rightarrow G, H$

$R6 = \{\underline{F}, G, H\}$

$R7 = \{\underline{B}, F\}$

There are no transitive dependencies in R3

Now, final set of relations are

{D, I, J}
{A, D, E}
{F, G, H}
{B, F}
{A, B, C}

Example 6: Apply first, second and third normal form on following relation.

Emp_Proj

EmpNo	Name	Dept.	Manager	ProjId	Proj Start_Dt	Location	Week
1.	Ashish	Accounts	John	A	15-10	Delhi	15
				B	11-10	Bangalore	12
				C	2-09	Hyderabad	9
2	Nidhi	Marketing	Smith	C	15-11	Bangalore	7

First Normal Form of Emp_Proj Relation

Primary key is EmpNo
ProjId, Proj_Start_Dt, Location and Week repeat for each value of EmpNo.
So, decompose the relation into following relations.
Employee (EmpNo, Name, Dept, Manager)
EmpProj1(EmpNo, Proj–Id, Proj_Start_Dt, Location, Week)

Second Normal Form
FD's in Employee Relation are:

EmpNo → Name, Dept, Manager

No partial dependencies exist in Employee relation since primary key is not composite.

FD's in relation EmpProj1 are:

{EmpNo, ProjId} → Proj_Start_Dt, Location, Week
ProjId → Proj_Start_Dt (this is a partial dependency)
So, decompose the relation EmpProj1
New relations are formed after decomposition are
EmpProj2 (EmpNo, ProjId, Location, Week)
Project (ProjId, Proj_Start_Dt)

Final set of relation after 2NF are

Employee (EmpNo, Name, Dept, Manager)
Employee2 (EmpNo, ProjId, Location, Week)
Project (ProjId, Proj_Start_Dt)

Third Normal Form

Transitive FD's exist in Employee Relation
EmpNo → Dept → Manager

So, decompose this relation. New relation formed are:

Employee2 (EmpNo, Name, Dept)
Department (Dept, Manager)

There are no transitive dependencies in EmpProj2 and Project
Final set of Relations after 3rd NF are:

Employee2 (EmpNo, Name, Dept)
Department (Dept, Manager)
Employee2 (EmpNo, ProjID, Location, Week)
Project (ProjId, Proj_Start_Dt)

Example 7: Consider a relation R with four attribute ABCD for the following set of FD do the following:

(a) Identify candidate key
(b) Identify the best normal form that satisfies
(c) If R is not in BCNF, decompose it into a set of BCNF relation that preserves dependencies

1. C → D, C → A, B → C

Solution. (a) Candidate key: B
(b) R is in 2NF
(c) R is not in BCNF because
C → D and C → A
To convert R into BCNF we decompose R into AC, BC and CD.

2. ABC → D, D → A

Solution. (a) Candidate key: ABC

(b) R is in 3NF

(c) ABCD is not in BCNF since
D → A and D is not a key. If we split up R as AD, BCD we cannot preserve dependency ABC → D. So there is no BCNF decomposition.

Example 8: Suppose we have a relation R=(A, B, C, D, E). FDs are:
(MDU, May 2010; GGSIPU, 2011)

(i) A→BC, (ii) CD→E (iii) B→D (iv) E→A

Decompose R into BCNF.

Solution: R = (A, B, C, D, E) with FD's

A→BC
CD→E
B→D
E→A

For BCNF we have the following conditions:

1. Relation should be in 2NF.
2. X→A, where X is a superkey and A is not a prime attribute.
So, first we find the key of relation.

$\{A^+\} \rightarrow ABCDE$
$\{CD^+\} \rightarrow CDEAB$
$\{B^+\} \rightarrow BD$
$\{E^+\} \rightarrow EABCD$

Now eliminate D from relation because B is not a superkey also decompose R in R_1 and R_2.

R1 (A, B, C, E)
R2 (B, D)

FD's in R1
A→BC

FD's in R2
B→D
C→E
E→A
FD CD→E is lost.

Example 9: Compute the closure of the following set F of FDs for relation schema R = (A, B, C, D, E)

(i) A→BC, (ii) CD→E, (iii) B→D, (iv) E→A

List the candidate keys of R. Also compute canonical cover of F_c

(GGSIPU 2012)

Solution:

A→BC (Given)
A→B
A→C (Decomposition Role)

For candidate key, compute the closure of attributes

$\{A^+\} = (A, B, C, D, E)$
$\{CD^+\} = \{C, D, E, A, B\}$ or $\{A, B, C, D, E\}$
$\{B^+\} = (B, D)$
$\{E^+\} = \{E, A, B, C, D\}$ or $\{A, B, C, D, E\}$

Also B→D
∴ BC→CD (angmentative)
So BC→ABCDE transitive [∵ CD → ABCDE]

∴ A, CD, E and BC are candidate keys of the relation.

To find canonical cover.

$F = \{A \rightarrow BC, CD \rightarrow E, B \rightarrow D, E \rightarrow A\}$

Step 1: $A \rightarrow B, A \rightarrow C, CD \rightarrow E, B \rightarrow D, E \rightarrow A$

Step 2:

- $A \rightarrow B$. Find $\{A\}^+$ without FD $A \rightarrow B$.
  $\{A\}^+ = \{AC\}$.
  Since B is not in A^+ ∴ $A \rightarrow B$ is not redundant
- $A \rightarrow C$. Find $[A]^+$ Without $A \rightarrow C$.
  $[A]^+ = \{ABD\}$.
  $A \rightarrow C$ is not redundant.
- $CD \rightarrow E$. Find $\{CD\}^+$ without $CD \rightarrow E$
  $\{CD\}+ = \{CD\}$.
  $CD \rightarrow E$ is not redundant.
- $B \rightarrow D$. Find $\{B\}+$ without $B \rightarrow D$
  $\{B^+\} = \{B\}$
  B is not redundant.
- $E \rightarrow A$. Find $\{E\}^+$ without $E \rightarrow A$.
  $\{E^+\} = \{E\}$
  E is not redundant

Step 3: For $CD \rightarrow E$

For C, find $\{D\}^+$ wrt $J = \{A \rightarrow B, A \rightarrow C, D \rightarrow E, B \rightarrow D, E \rightarrow A\}$
For C, find $\{D\}^+$ wrt $F = \{A \rightarrow B, A \rightarrow C, CD \rightarrow E, B \rightarrow D, E \rightarrow A\}$
$\{D\}^+$ under J ≠ $\{D\}^+$ under F. So C is not extraneous.

Similarly for D, $\{C\}^+$ under $J = \{A \to B, A \to C, C \to E, B \to D, E \to A\}$ is not qual to $\{C\}^+$ under $F = \{A \to B, A \to C, CD \to E, B \to D, E \to A\}$
$\therefore$ D is not extraneous.
Hence minimal cover of F is F
i.e. $\{A \to BC, CD \to E, B \to D, E \to A\}$

Example 10: Consider the relational schema.

.R = (A, B, C, D, E, F, G, H) with FDs

AB→C
BC→D
E→F
G→F
H→A
FG→H

Is decomposition of R into
R1 = (A, B, C, D), R2 = (A, B, C, E, F) and R3 = (A, D, F, G, H) is loss less?

Solution. Put symbol 'a' in the box for those elements which are present in he given set and put symbol 'b' in the box for those elements which are not present in the given set.

	A	B	C	D	E	F	G	H
R1	a_1	a_2	a_3	a_4	b_{15}	b_{16}	b_{17}	b_{18}
R2	a_1	a_2	a_3	a_{24}	a_5	a_6	b_{27}	b_{28}
R3	a_1	b_{32}	b_{33}	a_4	b_{35}	a_6	a_7	a_8

Now after applying FD $BC \to D$

New matrix would be

	A	B	C	D	E	F	G	H
R1	a_1	a_2	a_3	a_4	b_{15}	b_{16}	b_{17}	b_{18}
R2	a_1	a_2	a_3	a_4	a_5	a_6	b_{27}	b_{28}
R3	a_1	b_{32}	b_{33}	a_4	b_{35}	a_6	a_7	a_8

Since none of the row's is completely fill will all a's. Hence it is lossy-join decomposition.

Example 11: Assume that we wish to construct a database from a set of data items {A, B, C, D, E, F, G} and set F of FDs given by

(i) $BCD \to A$
(ii) $BC \to E$

(iii) $A \rightarrow F$
(iv) $F \rightarrow G$
(v) $C \rightarrow D$
(vi) $A \rightarrow G$

Start with table T containing all these attributes, and perform a lossless decomposition into two tables, T1 and T2, that make up a 2NF decomposition. List the keys for each table (T, T1, T2) and the FDs that lie in each table.

Solution. B and C are not on righthand side of any FD and therefore must be in any key, since a key functionally determins all other columns.

$BC \rightarrow A$ ... [From (i) and (v]
$BC \rightarrow E$ ... [(ii)]
$BC \rightarrow F$ ... [From (iii) by transitivity]
$BC \rightarrow D$ [From (v)]

Thus BC is a superkey for T which implies BC is the key.
Now D is not fully functionally dependent on BC, but dependent on C alone.
So we decompose the relation T into

$T1 = (\underline{B, C}, A, E, F, G)$ and
$T2 = (\underline{C}, D)$

Now $A \rightarrow F$, $F \rightarrow G$ are transitive dependencies. Thus T1 and T2 form a 2NF decomposition.

FDs in T1

$BC \rightarrow A$
$BC \rightarrow E$
$A \rightarrow F$
$F \rightarrow G$
$A \rightarrow G$

FDs in T2

$C \rightarrow D$

Example 12: Consider a universal relation R (A, B, C, D, E, F) and the set of functional dependencies.

(i) $A \rightarrow BCDEF$
(ii) $BC \rightarrow ADEF$
(iii) $B \rightarrow F$
(iv) $D \rightarrow E$

What is the key of R? Is the relations in 3NF? If not then convert it into 3NF.

Solution. To find the key of relation R we find the closure

$$\{A+\} = (A, B, C, D, E, F)$$
$$\{BC^{+}\} = (A, B, C, D, E, F)$$
$$\{B^{+}\} = (B, F)$$
$$\{D^{+}\} = (D, E)$$

A, and BC are the key of relation. First check whether the relation is in 2NF. FD $B \rightarrow F$ is a partial functional dependency, so relation is not in 2NF. To remove partial functional dependency we decompose the relation R into

$$R1 = (A, B, C, D, E)$$
$$R2 = (B, F)$$

For R1 the FD's are:

$$A \rightarrow BCDE$$
$$BC \rightarrow ADE$$
$$D \rightarrow E$$

For R2 FD's are:

$$B \rightarrow F$$

All the above FD satisfies condition of 2NF.
Now we check for 3NF.
There exist a transitive dependency in above FDs

$$A \rightarrow D$$
$$D \rightarrow E$$

So R1 is not in 3NF. To convert R1 into 3NF we decompose the relation R1 into

$$R1 \rightarrow (A, B, C, D)$$
$$R4 \rightarrow (D, E)$$

Now there are three relations after applying 3NF.

$$R1 \rightarrow (B, F)$$
$$R3 \rightarrow (A, B, C, D)$$
$$R2 \rightarrow (D, E)$$

Example 13: Find minimal set of functional dependencies from the following set: **(GGSIPU, 2005, 2010)**

$PQ \rightarrow R$
$PS \rightarrow Q$
$QS \rightarrow P$
$PR \rightarrow Q$
$S \rightarrow P$

Step 1: Every attribute should have only one attribute on right hand side.

$G = \{PQ \rightarrow R$

$PS \rightarrow Q$
$QS \rightarrow P$
$PR \rightarrow Q$
$S \rightarrow R\}$

Step 2: Remove redundant FDs

For $PQ \rightarrow R$ Compute $\{PQ\}^+$ wrt $\{PS \rightarrow Q, QS \rightarrow P, PR \rightarrow Q, S \rightarrow R\}$
$\{PQ\}^+ = \{PQ\}$. Since R is not in $\{PQ\}^+$ So remove nothing from G
$G = \{PQ \rightarrow R, PS \rightarrow Q, QS \rightarrow P, PR \rightarrow Q, S \rightarrow R\}$
For $PS \rightarrow Q$ Compute $\{PS\}^+ = \{PQ \rightarrow R, QS \rightarrow P, PR \rightarrow Q, S \rightarrow R\}$
$\{PS\}^+ = \{PSRQ\}$
Since Q is in $\{PS\}^+$ remove $PS \rightarrow Q$ from G. $QS \rightarrow P$
$G = \{PQ \rightarrow R, QS \rightarrow P, PR \rightarrow Q, S \rightarrow R\}$
For $QS \rightarrow P$. Compute $\{QS\}^+ = \{PQ \rightarrow R, QS \rightarrow P, PR \rightarrow Q, S \rightarrow P\}$
$\{QS\}^+ = \{QSRP\}$
Since P in $\{QS\}^+$ so remove $QS \rightarrow P$ from G
$G = \{PQ \rightarrow R, PR \rightarrow Q, S \rightarrow R\}$
For $PR \rightarrow Q$
$\{PR\}^+ = \{PR\}$
$G = \{PQ \rightarrow R, PR \rightarrow Q, S \rightarrow R\}$
For $S \rightarrow R$
$\{S\}^+ = \{S\}$
$G = \{PQ \rightarrow R, PR \rightarrow Q, S \rightarrow R\}$

Step 3: Remove redundant left hand side attributes.

$PQ \rightarrow R$
For P compute.
$\{Q\}^+$ wrt $\{Q \rightarrow R, PR \rightarrow Q, S \rightarrow R\} = \{QR\}$
Since Q^+ does not contain P, is QP is not redundant in $PQ \rightarrow R$.
For Q compute.
P^+ wrt $\{P \rightarrow R, PR \rightarrow Q, S \rightarrow R\} = \{PRQ\}$
P^+ contain Q, so Q is redundant in $PQ \rightarrow R$
For $PR \rightarrow Q$.
For P compute
R+ wrt $\{R \rightarrow Q, PQ \rightarrow R, S \rightarrow R\} = \{RQ\}$
P is not redundant in $PR \rightarrow Q$
For R compute
P^+ wrt $(P \rightarrow Q, PQ \rightarrow R, S \rightarrow R) = \{PQR\}$
P^+ contain R so R is redundant in $PR \rightarrow Q$.
$\therefore$ minimal closure set of given FDs are
$P \rightarrow R$
$P \rightarrow Q$
$S \rightarrow R$

Example 14: Consider the following two sets of functional dependencies.

$F = [A \rightarrow C, AC \rightarrow D, E \rightarrow AD, E \rightarrow H]$
$G = [A \rightarrow DC, E \rightarrow A]$

Check whether they are equivalent or not.

To show F and G are equivalent, we show that G is covered by F and F is covered by G. **(GGSIPU, 2010)**

I. G is covered by F.
$[A]^+ = \{A,C, D\}$ which covers $A \rightarrow CD$ in G.
$[E]^+ = \{E, A, D, H, C\}$ which covers $E \rightarrow AH$ in G

II. F is covered by G.
$[A]^+ = \{A, C, D\}$ which covers $A \rightarrow C$ in F.
$[A, C]^+ = \{A, C, D\}$ which covers $AC \rightarrow D$ in F.
$[F]^+ = \{E, A, H, C, D\}$ which covers $E \rightarrow AD$ and $E \rightarrow H$ in F.

Example 15: Consider the relation for published books.

Book (Book_title, Authorname, Book_type, Listprice, author_affil., Publisher)

Author_affil refers to the affiliation of author. Suppose the following dependencies exist.

Book_title → Publisher, Book_type.
Book_type → Listprice.
Ahthorname → Author_affil.

(a) What normal form is the relation in? Explain your answer.
(b) Apply normalization until you cannot decompose the relations further. State the reasons behind each decompositon. **(GGSIPU, 2009, 2011)**

Ans. (a) From the given FDs we infer that {Book_title, Authorname} should be the primary key for the relation Book as from these two attribute we can determine all other attributes.

Now Since,
{Book_title, Authorname} → Publisher, Book_type
and
Book_title → Publisher, Book_type.
∴ a partial dependency exist and hence not in 2NF.
So the relation is in 1 NF.

(b) Decomposition to 2NF
Book 1 (Book_title, Authorname)
Book 2 (Book_title, Publisher, Book_type, Listprice)
Book 3 (Authorname, Author_affil).

There is no partial dependencies in above relation and hence in 2NF
Decompositon to 3 NF

We need to eliminate transitive dependency from Book2 relation.

Book_title → Book_type
Book_type → Listprice. [Transitive dependency]

After decomposition relation are

Book 1 (Book_title, Authorname)
Book 3 (Authorname, Author_affil).
Book 4 (Book_title, Publisher, Book_type)
Book 5 (Book_type, Listprice).

Example 16: Relation R (A, B, C, D, E, F) satisfies the following dependencies AB → C, C → A, BC → D, ACD → B, BE → C, CE → FA, CF → BD, D → EF Find out the closure of FD's **(GGSIPU, 2011)**

i) AB → C

$\{A, B\}^+ = \{A, B\}$
$= \{A, B, C\}$ AB → C.
$= \{A, B, C, D\}$ BC → D
$= \{A, B, C, D, E, F\}$ D → EF
$= \{A, B, C, D. E, F\}$

(ii) C → A

$\{C\}^+ = \{C\}$
$= \{C, A\}$ C → A
$= \{A, C\}$

(iii) BC → D

$\{B, C\}^+ = \{B, C\}$
$= \{B, C, D\}$ BC → D
$= \{A, B, C, D\}$ C → A
$= \{A, B, C, D, E, F\}$ D → EF
$= \{A, B, C, D, E, F\}$

(iv) ACD → B

$\{A, C, D\}^+ = \{A, C, D)$
$= \{A, C, D, B\}$ ACD → B
$= \{A, B, C, D, E, F\}$ D → EF
$\{A, B, C, D, E, F\}$

(v) BE → C

$\{B, E\}^+ = \{B, E\}$
$= \{B, C, E\}$ BE → C
$= \{B, C, D, E\}$ BC → D
$= \{A, B, C, D, E\}$ C → A
$= \{A, B, C, D, E, F\}$ D → EF
$= \{A, B, C, D, E, F\}$

(vi) $CE \rightarrow FA$

$\{C, E\}^+ = \{C, E\}$
$= \{C, E, A, F\}$ $CE \rightarrow FA$
$= \{A, B, C, D, E, F\}$ $CF \rightarrow BD$
$= \{A, B, C, D, E, F\}$

(vii) $CF \rightarrow BD$

$\{C, F\}^+ = \{C, F\}$
$= \{C, F, B, D\}$ $CF \rightarrow BD$
$= \{B, C, D, E, F\}$ $D \rightarrow EF$
$= \{A, B, C, D, E, F\}$ $C \rightarrow A$
$= \{A, B, C, D, E, F\}$

(viii) $D \rightarrow EF$

$\{D\}^+ = \{D\}$
$= \{D, E, F\}$ $D \rightarrow EF$
$= \{D, E, F\}$

Example 17: Differentiate between 3 NF and BCNF. **(GGSIPU 2009)**

Ans. • **3NF:** A relation R is said to be in 3NF if it is in 2NF and no non prime attribute is transitively dependent on primary key of R.

BCNF: A relation R is said to be in BCNF if and only if every determinant is a candidate key. Here determinant can be simple or composite.

- **3NF:** 3NF allowes a FD $A \rightarrow B$, if B is a primary key and is not a candidate key.
  BCNF: If $A \rightarrow B$ is a CD then A must be a candidate key.
- **3NF:** is not a stronger form of BCNF. Every relation in 3 NF is not necessary also in BCNF.
  BCNF: It is stronger form of 3NF. Every relation in BCNF is also in 3NF.

Example 18: Prove that any relational schema with two attributes is in BCNF. **(GGSIPU 2007)**

Ans. To prove this, consider a relational schema R with two attributes {A, B). Only two non trivial functional dependencies are possible corresponding to relation R which is $A \rightarrow B$ and $B \rightarrow A$. Now a relation is in BCNF if for every FD that exists in R, determinants should be candidate key. There are four cases possible which are:

i) Both FDs $A \rightarrow B$ and $B \rightarrow A$ do not hold for R: This means the key for R should be {A,B} which implies R is in BCNF.
ii) Only $A \rightarrow B$ holds for R: This means the key for R is A which implies R is in BCNF.
iii) Only $B \rightarrow A$ holds for R: This means the key for R is B which implies R is in BCNF.
iv) Both FDs $A \rightarrow B$ and $B \rightarrow A$ holds for R: This means the key for R can be A or B which implies R is in BCNF.

Hence it is proved that any relational schema with two attributes is in BCNF.

Example 19: Explain why 4NF is more desirable than BCNF.

Ans. Consider the following set of FDs are in BCNF as all of them are trivial but not in 4NF.

$$A \rightarrow\rightarrow BC$$
$$B \rightarrow\rightarrow CD$$
$$E \rightarrow\rightarrow AD$$

4NF is more desirable then BCNF because it reduces the repetition of information. In above schema we see that decomposition into 4NF does not loose information provided that lossless join decomposition is used yet redundancy is reduced.

Example 20: Consider the relation R(X, Y, Z) given below.

X	Y	Z
6	4	2
6	6	8
6	4	8

Which of these FD do not hold for relation R?

$$Y \rightarrow X$$
$$Z \rightarrow Y$$
$$XY \rightarrow Z$$

Ans. FD Y→X holds for R as for any value in Y we have same value in X.

FD Z→Y does not hold for R since second and third tuples both have 8 for Z but different value for Y.

FD XY→Z does not hold for R since first and third tuples both have 6 and 4 for X and Y but different values for Z.

Example 21: Given a relation R(ABC) with set F of FDs {A→B, B→C, C→A}. If we decompose R into R1(AB), R2(BC), then find whether it is dependency preserving or not.

Ans. Given $F = \{A \rightarrow B, B \rightarrow C, C \rightarrow A\}$.

Compute $F^+ = \{A \rightarrow B, B \rightarrow C, C \rightarrow A,$
$B \rightarrow A, (B \rightarrow C \text{ and } C \rightarrow A)$
$C \rightarrow B, (C \rightarrow A \text{ and } A \rightarrow B)$
$A \rightarrow C\}$ $(A \rightarrow B \text{ and } B \rightarrow C)$

Now find $(F_{AB} \cup F_{BC})$

$$F_{AB} = \{A \rightarrow B, B \rightarrow A\}$$
$$F_{BC} = \{B \rightarrow C, C \rightarrow B\}$$

$$F_{AB} \cup F_{BC} = \{A \rightarrow B, B \rightarrow A, B \rightarrow C, C \rightarrow B\}$$
$$(F_{AB} \cup F_{BC})^+ = \{A \rightarrow B, B \rightarrow A, B \rightarrow C, C \rightarrow B, A \rightarrow C, C \rightarrow A\}$$
$$(F_{AB} \cup F_{BC})^+ = F^+$$

Hence the decomposition is dependency preserving.

Example 22: a) Why certain functional dependencies are called non trival FDs?

b) In designing a relational database, why might we choose a non BCNF design? **(GGSIPU 2009)**

Ans. a)Some functional dependencies are call trival functional dependencies because they are satisfied by all relations.

b) BCNF is not always dependency preserving. So some other normal form can be chosen like 3NF so that checking of dependencies becomes easier during updates. This avoid joins to check dependencies and increase performance.

Example 23: Find out the minimal cover for set F of FD, $A \rightarrow BC$, $B \rightarrow C$, $A \rightarrow B$, $AB \rightarrow C$, $AC \rightarrow D$.

Step 1: Apply decomposition rule to obtain equivalent set with single attribute on right side.

$$F = \{A \rightarrow B, A \rightarrow C, B \rightarrow C, A \rightarrow B, AB \rightarrow C, AC \rightarrow D\}$$

Step 2: Remove any redundant FDs

- FD $A \rightarrow C$ appears twice in F so remove one of these FD.
  $F = \{A \rightarrow B, B \rightarrow C, A \rightarrow C, AB \rightarrow C, AC \rightarrow D\}$
- Consider $A \rightarrow B$, compute $\{A\}^+$ without FD $A \rightarrow B$
  $\{A\}^+ = \{ACD\}$
  Since B is not in $\{A\}^+$, FD $A \rightarrow B$ is essential.
  $F = \{A \rightarrow B, B \rightarrow C, A \rightarrow C, AB \rightarrow C, AC \rightarrow D\}$
- Consider $A \rightarrow C$, compute $\{A\}^+$ without FD $A \rightarrow C$
  $\{A\}^+ = \{ABCD\}$
  Since C is in $\{A\}^+$, FD $A \rightarrow C$ is not essential. So remove FD $A \rightarrow C$
  $F = \{A \rightarrow B, B \rightarrow C, AB \rightarrow C, AC \rightarrow D\}$
- Consider $B \rightarrow C$, compute $\{B\}^+$ without FD $B \rightarrow C$
  $\{B\}^+ = \{B\}$
  Since C is not in $\{B\}^+$, FD $B \rightarrow C$ is essential
  $F = \{A \rightarrow B, B \rightarrow C, AB \rightarrow C, AC \rightarrow D\}$
- Consider $AB \rightarrow C$, compute $\{AB\}^+$ without FD $AB \rightarrow C$
  $\{AB\}^+ = \{ABCD\}$
  Since C is in $\{AB\}^+$, FD $AB \rightarrow C$ is not essential. So remove $AB \rightarrow C$
  $F = \{A \rightarrow B, B \rightarrow C, AC \rightarrow D\}$
- Consider $AC \rightarrow D$, compute $\{A\}+$ without FD $AC \rightarrow D$
  $\{AC\}^+ = \{ACB\}$
  Since D is not in $\{AC\}^+$, FD $AC \rightarrow D$ is essential.

$$F = \{A \rightarrow B, B \rightarrow C, AC \rightarrow D\}$$

Step 3: Remove extraneous attribute if any for FD they have more than one attribute on left side

$$F = \{A \rightarrow B, B \rightarrow C, AC \rightarrow D\}$$

- Consider $AC \rightarrow D$
  For C compute $\{A\}^+$ under set $J = \{A \rightarrow B, B \rightarrow C, A \rightarrow D\}$
  $\{A\}^+ = \{ABCD\}$
  For B compute $\{A\}^+$ under set $F = \{A \rightarrow B, B \rightarrow C, AC \rightarrow D\}$
  $\{A\}^+ = \{ABCD\}$
  $\{A\}^+$ under J is not equal to $\{A\}^+$ under F,therefore C is an extraneous attribute.
  So replace $AC \rightarrow D$ with $A \rightarrow D$.
  $F = \{A \rightarrow B, B \rightarrow C, A \rightarrow D\}$

Step 4: Using union rule combine FDs having same left side.

$$M = \{A \rightarrow BD, B \rightarrow C\}$$